[0033] **FIG. 13** is a top view of a sensor arrangement of a touch pad, in accordance with another embodiment of the present invention.

[0034] **FIG. 14** is a top view of a sensor arrangement of a touch pad, in accordance with another embodiment of the present invention.

[0035] **FIG. 15** is a top view of a sensor arrangement of a touch pad, in accordance with another embodiment of the present invention.

DETAILED DESCRIPTION OF THE INVENTION

[0036] The present invention will now be described in detail with reference to a few preferred embodiments thereof as illustrated in the accompanying drawings. In the following description, numerous specific details are set forth in order to provide a thorough understanding of the present invention. It will be apparent, however, to one skilled in the art, that the present invention may be practiced without some or all of these specific details. In other instances, well known process steps have not been described in detail in order not to unnecessarily obscure the present invention.

[0037] **FIG. 2** is a diagram of a computing system **20**, in accordance with one embodiment of the present invention. The computing system **20** includes at least a user interface **22** and a host device **24**. The user interface **22** is configured to provide control information for performing actions in the host device **24**. By way of example, the actions may include making selections, opening a file or document, executing instructions, starting a program, viewing a menu, and/or the like. The actions may also include moving an object such as a pointer or cursor on a display screen of the host device **24**. Although not shown in **FIG. 2**, the user interface **22** may be integrated with the host device **24** (within the same housing) or it may be a separate component (different housing).

[0038] The user interface **22** includes one or more touch buttons **34**, a touch pad **36** and a controller **38**. The touch buttons **34** generate button data when a user places their finger over the touch button **34**. The touch pad, on the other hand, generates position data when a user places their finger (or object) over the touch pad **36**. The controller **38** is configured to acquire the button data from the touch buttons **34** and the position data from the touch pad **36**. The controller is also configured to output control data associated with the button data and/or position data to the host device **24**. In one embodiment, the controller **38** only outputs control data associated with the touch buttons when the button status has changed. In another embodiment, the controller **38** only outputs control data associated with the touch pad when the position data has changed. The control data, which may include the raw data (button, position) or some form of thereof, may be used to implement a control function in the host device **24**. By way of example, the control data may be used to move an object on the display **30** of the host device **24** or to make a selection or issue a command in the host device **24**.

[0039] The touch buttons **34** and touch pad **36** generally include one or more sensors capable of producing the button and position data. The sensors of the touch buttons **34** and touch pad **36** may be distinct elements or they may be grouped together as part of a sensor arrangement, i.e., divided into sensors for the touch buttons **34** and sensors for the touch pad **36**. The sensors of the touch buttons **34** are configured to produce signals associated with button status (activated, not activated). For example, the button status may indicate button activation when an object is positioned over the touch button and button deactivation at other times (or vice versa). The sensors of the touch pad **36** are configured produce signals associated with the absolute position of an object on or near the touch pad **36**. In most cases, the sensors of the touch pad **36** map the touch pad plane into native or physical sensor coordinates **40**. The native sensor coordinates **40** may be based on Cartesian coordinates or Polar coordinates (as shown). When Cartesian, the native sensor coordinates **40** typically correspond to x and y coordinates. When Polar (as shown), the native sensor coordinates typically correspond to radial and angular coordinates (r, θ). By way of example, the sensors may be based on resistive sensing, surface acoustic wave sensing, pressure sensing (e.g., strain gauge), optical sensing, capacitive sensing and the like.

[0040] In one embodiment, the user interface **22** includes a sensor arrangement based on capacitive sensing. The user interface **22** is therefore arranged to detect changes in capacitance as a finger moves, taps, or rests on the touch buttons **34** and touch pad **36**. The capacitive touch assembly is formed from various layers including at least a set of labels, a set of electrodes (sensors) and a printed circuit board (PCB). The electrodes are positioned on the PCB, and the labels are position over the electrodes. The labels serve to protect the electrodes and provide a surface for receiving a finger thereon. The label layer also provides an insulating surface between the finger and the electrodes. As should be appreciated, the controller **38** can determine button status at each of the touch buttons **34** and position of the finger on the touch pad **36** by detecting changes in capacitance. In most cases, the controller **38** is positioned on the opposite side of the PCB. By way of example, the controller **38** may correspond to an application specific integrated circuit (ASIC), and it may operate under the control of Firmware stored on the ASIC.

[0041] Referring to the controller **38**, the controller **38** is configured to monitor the sensors of the touch buttons **34** and touch pad **36** and decide what information to report to the host device **24**. The decision may include filtering and/or conversion processes. The filtering process may be implemented to reduce a busy data stream so that the host device **24** is not overloaded with redundant or non-essential data. By way of example, a busy data stream may be created when multiple signals are produced at native sensor coordinates **40** that are in close proximity to one another. As should be appreciated, processing a busy data stream tends to require a lot of power, and therefore it can have a disastrous effect on portable devices such as media players that use a battery with a limited power supply. Generally speaking, the filtering process throws out redundant signals so that they do not reach the host device **24**. In one implementation, the controller **38** is configured to only output a control signal when a significant change in sensor signals is detected. A significant change corresponds to those changes that are significant, as for example, when the user decides to move his/her finger to a new position rather than when the user's finger is simply resting on a spot and moving ever so slightly because of finger balance (toggling back and forth). The filter process